Java

المحاضرة الخامسة عشر — الأحد — 26/11/2017

الصفوف classes

الصف: هو أداة برمجية تمكننا من تعريف نوع جديد بالإضافة للعمليات على هذا النوع وكذلك إمكانية تقييد الوصول إلى محتويات هذا النوع

* العمليات على النوع هي الطرق (الدوال) Method
* تقييد الوصول يتم عن طريق محددات الوصول (الافتراضي - العام - الخاص - المحمي)

تعريف صف برمجياً:

```
class اسم الصف {
    [معطيات أعضاء]
    [دوال أعضاء]
}
```

اسم الصف: هو اسم اختياري على أن يحترم قواعد التسمية وتم الاتفاق بين المبرمجين في البرمجة غرضية التوجه على أن يبدأ اسم الصف بحرف كبير واسم الطريقة بحرف صغير (اتفاقية وليست قاعدة)

المتحولات الأعضاء: متحولات من أنواع بسيطة (مثل ... int, double) أو من أنواع معرفة مسبقاً أي أغراض (أستطيع أن استخدم غرض من صف معرف مسبقاً كمعطى عضو داخل صف)

مثال:

```
class Complex {
    double x;
    double y;
}
```

* حتى نتمكن من استخدام أي نوع يجب تعريف أو التصريح عن متحولات منه باستخدام قاعدة تعريف المتحولات البسيطة تقريباً

واسم المتحول مزاغ النوع

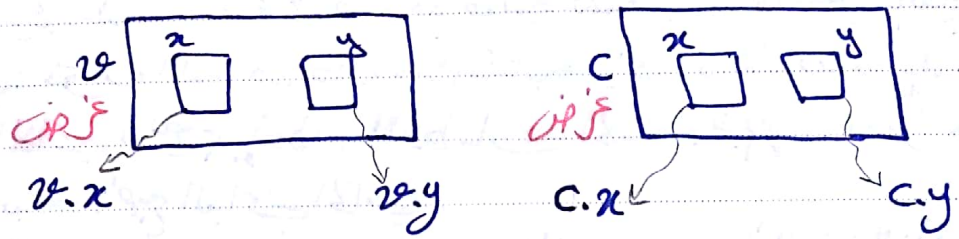
سيصبح متغيراً أو متحول

واسم متحول مزاغ اسم صف

سيصبح غرض object

تصريح عن c,z2 من النوع Complex

Complex c, z2;



* في الذاكرة فقط المطيات الأحضاء التي تتخزن أما الطرف فلا تتخزن في الذاكرة
شرح: كل غرض من النوع Complex رح يحوي على x و y أنا ما بقدر أوصل لـ x الغرض قبل ما أوصل للغرض نفسه يلي بيحويه وهاد الكلام عبرناعنه بالكتابة c.x , c.y أو z2.x, z2.y

* حق استطيع استخدام متحول من نوع صف أي غرض يجب ان يصرح عنه (أي كتابة النوع ومن ثم الاسم) ويجب ان يكون مبني أي مجوز له مكان في الذاكرة ومعطى قيم ابتدائية لكل معطى عضو

* الحجز في الذاكرة كل لغات البرمجة اتفقت على كلمة new للتعبير عن حجز جديد في الذاكرة والقاعدة المتبعة في كل لغات البرمجة هي كلمة new يتبعها النوع لكي يتم تحديد الحجم المراد حجزه.

بناء الغرض (object) يتم عن طريق الباني

الباني constractor: هو دالة (طريقة) خاصة ليس لها نوع ارجاع وتحمل اسم الصف نفسه ومهمتها اعطاء قيم ابتدائية للمطيات الأحضاء

الفرق بين دالة (function) وبين طريقة (method)

الدالة بشكل كامل لها الشكل: نوع ارجاع اسم قائمة وسطاء وجسم

الطريقة: هي دالة كتبت داخل صف

في لغة Java جميع الدوال هي طرق لأنه الـ Java هي لغة نقية

بالعودة لمثال الـ Complex:

```
class Complex{
  double x;
  double y;
Complex(){ x=0; y=0;}  الباني الخالي
  Complex(double a, double b){ x=a; y=b;}
}
```

الباني الخالي: يكتب بدون وسطاء يعطي المعطيات الأعضاء القيم الافتراضية (كتابة)
يكتب عندما لا يكون لدي حاجة لإعطاء قيم فعلية للمعطيات وأكتفي بالقيم الافتراضية
الباني الافتراضي: تولده اللغة في حال عدم كتابة باني داخل الصف له نفسه
شكل الباني الخالي ويقوم بإعطاء المعطيات الأعضاء قيمها الافتراضية
إذا قمنا بكتابة أصبح الباني الخالي.

* يمكن أن نكتب أكثر من دالة بنفس الاسم «مبدأ التحميل الزائد للطرق محقق»
* لكل دالة توقيع أو تروسية يميزها التوقيع يتكون من اسم الدالة مع قائمة وسطائها
* حتى نتمكن من كتابة دالتين في نفس البرنامج يجب أن تختلفا بالتوقيع أي أما أن تغير الاسم أو تغير الوسطاء (العدد - النوع) وإذا كانت الوسطاء من أنواع مختلفة نستطيع تغيير ترتيبهم.
* لو أردنا كتابة باني في صف الـ Complex ليبني الأعداد العقدية التي تحوي القسم الحقيقي فقط وأيضاً كتابة باني لبناء الأعداد العقدية التي تحوي القسم التخيلي فقط عندها:

الباني الأول:
```
Complex (double a){
x=a; y=0;}
```
الباني الثاني:
```
Complex(double b){
x=0; y=b;}
```

لا يمكن وجودهما مع بعض في نفس الصف بهذا الشكل لأن هذه الكتابة ليست تحميل زائد للطرق. لحل المشكلة نفرض محمول من نوع آخر غير الـ double وهو بمثابة محمول دلالي:
```
Complex(double a, int z){
if(z==0){x=a; y=0;}
else{x=0; y=a;}
}
```
قد احتاج أكثر من باني للصف لأن المعلومات لدي يمكن أن تأخذ أشكال مختلفة وفي ممكن أن تبنى بقيم ثابتة موجودة ومعروفة ويمكن أن تبنى حسب رغبة المستخدم.

Complex c; تصريح
c=new Complex(); بناء
Complex c=new Complex(); ⇐

انتهت المحاضرة